c) SEQUÊNCIA SIMPLES

PORTUGOL

```
<comando 1> ;
<comando 2> ;
    .
    .
    .
<comando n> ;
```

PASCAL

```
<comando 1> ;
<comando 2> ;
    .
    .
    .
<comando n> ;
```

A função do ; (ponto e vírgula) no PORTUGOL é o da indicação de fim de ação. No PASCAL o ; indica a separação de comandos.

d) ALTERNATIVA (simples e composta)

PORTUGOL

```
se <condição>
|     então <lista de comandos> ;
fim se;

se <condição>
|     então <lista de comandos1> ;
|     senão <lista de comandos2> ;
fim se;
```

PASCAL

```
if <expressão1>
    then <comando>
if <expressão>
    then <comando1>
    else <comando2>
```

Observe que depois da palavra *then* ou da palavra *else* só podemos ter *um comando*. Se houver mais de um comando, será necessário utilizar um *comando composto*, isto é, colocar os comandos entre as palavras *begin* e *end*.

Como exemplo, para passar para o PASCAL:

```
se A ≤ B
|  então  A ← A + 2;
|  senão  B ← B - 1;
|         N ← M;
fim se;
```

teríamos:

```
if A < = B
  then A: = A + 2
  else begin
             B: = B - 1;
             N: = M
       end
```

Observe que antes do *else* não pode aparecer um ponto e vírgula (;) já que o comando ainda não terminou. Não é necessário colocar ; antes do *end* (pela sintaxe), embora, por questões de estilo, seja aceito na maioria dos compiladores PASCAL.

Ao passar para o PASCAL:

```
se A ≠ B
|  então A ← A + 2;
|        B ← 3;
fim se;
```

teríamos:

```
if A <> B
  then begin
             A: = A + 2;
             B: = 3
       end
```

Observe que ≠ é mapeado por <>.

Considerando o trecho de programa PASCAL a seguir, que valores terão *B* e *C* depois de executados os comandos?

```
B: = 0;
C: = 0;
P: = 3;
if P < 3 then;  begin
                       B: = 5;
                       C: = 10
                end;
B: = B + 1;
C: = C + 1;
```

Solução:

B terá o valor 6 e *C* terá o valor 11. Observe que a identação no PASCAL não é reconhecida pelo compilador e o trecho acima equivale a: